

Locally Stable Matching with General Preferences*

Martin Hoefer

Lisa Wagner

Abstract

We study stable matching problems with locality of information and control. In our model, each player is a node in a fixed network and strives to be matched to another player. A player has a complete preference list over all other players it can be matched with. Players can match arbitrarily, and they learn about possible partners dynamically based on their current neighborhood. We consider convergence of dynamics to locally stable matchings – states that are stable with respect to their imposed information structure in the network. In the two-sided case of stable marriage in which existence is guaranteed, we show that reachability becomes NP-hard to decide. This holds even when the network exists only among one partition of players. In contrast, if one partition has no network and players remember a previous match every round, reachability is guaranteed and random dynamics converge with probability 1. We characterize this positive result in various ways. For instance, it holds for random memory and for cache memory with the most recent partner, but not for cache memory with the best partner. Also, it is crucial which partition of the players has memory. Finally, we present a variety of results for centralized computation of locally stable matchings, e.g., computing maximum locally stable matchings in the two-sided case and deciding and characterizing existence in the general roommates case.

*Department of Computer Science, RWTH Aachen University, Germany. {mhoefer,lwagner}@cs.rwth-aachen.de

1 Introduction

Matching problems form the basis for a variety of assignment and allocation tasks encountered in computer science, operations research, and economics. A prominent and popular approach in all these areas is *stable matching*, as it captures aspects like distributed control and rationality of participants that arise in many assignment problems today. A variety of allocation problems in markets can be analyzed within the context of two-sided stable matching, e.g., the assignment of jobs to workers [3, 10], organs to patients [18], or general buyers to sellers. In addition, stable marriage problems have been successfully used to study distributed resource allocation problems in networks [2, 5, 13].

In this paper, we consider a game-theoretic model for matching with distributed control and information. Players are rational agents embedded in a (social) network and strive to find a partner for a joint relationship or activity, e.g., to do sports, write a research paper, exchange data etc. Such problems are of central interest in economics and sociology, and they act as fundamental coordination tasks in distributed computer networks. Our model extends the stable marriage problem, in which we have sets U and W of men and women. Each man (woman) can match to at most one woman (man) and has a complete preference list over all women (men). Each player would rather be matched than unmatched. Given a matching M , a *blocking pair* is a man-woman pair such that both improve by matching to each other and leaving their current partner (if any). A matching without any blocking pair is a *stable matching*.

A central assumption in stable marriage is that every player knows all players it can match to. In reality, however, players often have limited information about their matching possibilities. For instance, in a large society we would not expect a man to match up with any other woman immediately. Instead, there exist restrictions in terms of knowledge and information that allow some pairs to match up directly, while others would have to get to know each other first before being able to start a relationship. We incorporate this aspect by assuming that players are embedded in a fixed network of *links*. Links represent an enduring knowledge relation that is not primarily under the control of the players. Depending on the interpretation, links could represent, e.g., family, neighbor, colleague or teammate relations. Each player strives to build one *matching edge* to a partner. The set of links and edges defines a dynamic information structure based on *triadic closure*, a standard idea in social network theory: If two players have a common friend, they are likely to meet and learn about each other. Translated into our model this implies that each player can match to partners in its 2-hop neighborhood of the network of matching edges and links. Then, a *local blocking pair* is a blocking pair of players that are at hop distance at most 2 in the network. Consequently, a *locally stable matching* is a stable matching without local blocking pairs. Local blocking pairs are a subset of blocking pairs. In turn, every stable matching is a locally stable matching, because it allows no (local or global) blocking pairs. Thus, one might be tempted to think that locally stable matchings are easier to find and/or reach using distributed dynamics than ordinary stable matchings. In contrast, we show in this paper that locally stable matchings have a rich structure and can behave quite differently than ordinary stable matchings. Our study of locally stable matching with general preferences significantly extends recent work on the special case of correlated or weighted matching [8], in which preferences are determined by benefits for each matching edge.

Contribution For most of the paper, we concentrate on the important two-sided scenario of stable marriage, in which a (locally) stable matching is always guaranteed to exist. Our primary interest is to characterize convergence properties of iterative round-based dynamics with distributed control, in which in each round a local blocking pair is resolved. We focus on the REACHABILITY problem:

Given a game and an initial matching, is there a sequence of local blocking pair resolutions leading to a locally stable matching? In Section 3 we see that there are cases, in which a locally stable matching might never be reached. This is in strong contrast to the case of weighted matching, in which it is easy to show convergence of every sequence of local blocking pair resolutions with a potential function [13]. In fact, it is NP-hard to decide REACHABILITY, even if the network exists only among one partition of players. Moreover, there exist games and initial matchings such that *every* sequence of local blocking pairs terminating in a locally stable matching is exponentially long. Hence, REACHABILITY might even be outside NP. If we need to decide REACHABILITY for a given initial matching *and a specific locally stable matching to be reached*, the problem is even NP-hard for correlated or weighted matching, where preferences are determined by edge weights.

Given our NP-hardness results, in Section 4 we concentrate on a more restricted class of games in which one partition has no internal links, i.e., links exist only between partitions and among the other partition. This is a natural assumption when considering objects that do not generate knowledge about each other, e.g., when matching resources to networked nodes or users, where initially resources are only known to a subset of users. Here we characterize the impact of memory on distributed dynamics. For *recency memory*, each player remembers in every round the *most recent partner* that is different from the current one. With recency memory, REACHABILITY is always true, and for every initial matching there exists a sequence of polynomially many local or remembered blocking pairs leading to a locally stable matching. In fact, we only need the partition without internal links to have recency memory. If, in contrast, only the other partition has recency memory, REACHABILITY becomes again NP-hard. The same hardness holds for *quality memory* if all players from both partitions remember their *best partner*. This formally supports the intuition that recency memory is more powerful than quality memory, as the latter can be easily misled in the course of a dynamic process. This provides a novel distinction between recency and quality memory that was not known in previous work.

Our positive results for recency memory in Section 4 imply that if we pick local blocking pairs uniformly at random in each step, we achieve convergence with probability 1. This can also be guaranteed for *random memory* if in each round each player remembers one of his previous matches chosen uniformly at random. In fact, for random memory this result holds even in general when links exist among or between both partitions. However, using known results on stable marriage with full information [1], convergence time can be exponential with high probability, independently of any memory.

In Sections 5.1 and 5.2 we also treat more centralized aspects of locally stable matching to highlight their different nature compared to ordinary stable matchings. A fundamental observation that motivates our results in Section 5.1 is that – in contrast to ordinary stable matchings – two locally stable matchings can have different size, and we consider the natural problem of finding a maximum locally stable matching. While a simple 2-approximation algorithm exists, we can show a non-approximability result of $1.5 - \varepsilon$ under the unique games conjecture. Finally, in Section 5.2 we consider the roommates problem, in which players can match arbitrarily to other players. In this case, we show that – in contrast to ordinary stable matchings – deciding existence of locally stable matchings is NP-complete. We also provide some initial results on characterizing the existence of locally stable matchings using linear programming.

Related Work Locally stable matchings were introduced by Arcaute and Vassilvitskii [3] in a two-sided job-market model, in which links exist only among one partition. The paper uses strong uniformity assumptions on the preference lists and addresses the lattice structure for stable matchings and a local Gale-Shapley algorithm. More recently, we studied locally stable matching

with correlated preferences in the roommates problem, where arbitrary pairs of players can be matched [8]. Using a potential function argument, REACHABILITY is always true and convergence guaranteed. Moreover, for every initial matching there is a polynomial sequence of local blocking pairs that leads to a locally stable matching. The expected convergence time of random dynamics, however, is exponential. If we restrict to resolution of pairs with maximum benefit, then for random memory the expected convergence time becomes polynomial, but for recency or quality memory convergence time remains exponential, even if the memory is of polynomial size.

For ordinary two-sided stable matching there have been a wide variety of works on various aspects, e.g., many-to-many matchings, ties, incomplete lists, etc. For an introduction to the topic we refer the reader to several books in the area [6, 15]. Theoretical work on convergence issues in ordinary stable marriage has focused on better-response dynamics, in which players sequentially deviate to blocking pairs. It is known that for stable marriage these dynamics can cycle [12]. On the other hand, REACHABILITY is always true, and for every initial matching there exists a sequence of polynomially many steps to a stable matching [16]. However, if blocking pairs are chosen uniformly at random at each step, convergence time is exponential [1].

For the more general roommates problem, in which every pair of players can be matched, stable matchings can be absent. There are algorithms to decide existence and compute stable matchings in polynomial time if they exist [9]. Some of these algorithms rely on a tight characterization of stable matchings in terms of linear programming [19].

Recently, the stable marriage problem with ties and incomplete lists has attracted interest. In contrast to ordinary stable matching, this case yields stable matchings of different sizes. A straightforward objective for a central planner is to match as many pairs as possible in a stable fashion. This problem was shown to be APX-hard in [7]. The problem has generated a significant amount of research interest over the past decade, and the currently best results are a 1.5-approximation algorithm [11, 14] and $(4/3 - \varepsilon)$ -hardness under the unique games conjecture [20].

2 Preliminaries

2.1 Our Model

A *network matching game* (or *network game*) consists of a (social) *network* $N = (V, L)$, where V is a set of vertices representing *players* and $L \subseteq \{\{u, v\} | u, v \in V, u \neq v\}$ is a set of fixed *links*. A set $E \subseteq \{\{u, v\} | u, v \in V, u \neq v\}$ defines the *potential matching edges*. A *state* is a matching $M \subseteq E$ such that for each $v \in V$ we have $|\{e \in M, v \in e\}| \leq 1$. An edge $e = \{u, v\} \in M$ provides utilities $b_u(e), b_v(e) > 0$ for u and v , respectively. If for every $e \in E$ we have $b_u(e) = b_v(e) = b(e) > 0$, we speak of *correlated preferences* or a *correlated network game*. Otherwise, we will assume that each player has a ranking \succ over its possible matching partners and the utility of e for u is given by the rank u assigns to v . If we can divide V into two disjoint sets U and W such that $E \subseteq \{\{u, w\} | u \in U, w \in W\}$, we call the game *bipartite*. We will focus on this case in Sections 3 and 4. Note that this does not imply that N has to be bipartite. If further the vertices of U are isolated in N , we term the game a *job-market game* for consistency with [3, 8].

To describe stability in network matching games, we assume players u and v are *accessible* in state M if they have a distance of at most 2 in the graph $G = (V, L \cup M)$. A state M has a *local blocking pair* $e = \{u, v\} \in E$ if u and v are accessible and are each either unmatched in M or matched through an edge e' such that e' serves a strictly smaller utility than e . Thus, in a local blocking pair both players can strictly increase their utility by generating e (and possibly dismissing some other edge thereby). A state M that has no local blocking pair is a *locally stable matching*.

Most of our analysis concerns iterative round-based dynamics, where in each round we pick one local blocking pair, add it to M , and remove all edges that conflict with this new edge. We call one such step a *local improvement step*. With *random dynamics* we refer to the process when in each step the local blocking pair is chosen uniformly at random from the ones available. A local blocking pair $\{u, v\}$ that is resolved must be connected by some distance-2 path (u, w, v) in M before the step. This path can consist of two links, or of exactly one link and one matching edge. In the latter case, let w.l.o.g. $\{u, w\}$ be the matching edge. As u can have only one matching edge, the local improvement step will delete $\{u, w\}$ to create $\{u, v\}$. For simplicity, we will refer to this fact as "an edge moving from $\{u, w\}$ to $\{u, v\}$ " or " u 's edge moving from w to v ".

In subsequent sections we will assume that players have memory that allows to "remember" one matching partner from a former round. In this case, a pair $\{u, v\}$ of players becomes accessible not only by a distance-2 path in G , but also when u appears in the memory of v . Hence, in this case a local blocking pair can be based solely on access through memory. For *random memory*, we assume that in every round each player remembers a previous matching partner chosen uniformly at random. For *recency memory*, each player remembers the last matching partner that is different from the current partner. For *quality memory*, each player remembers the best previous matching partner.

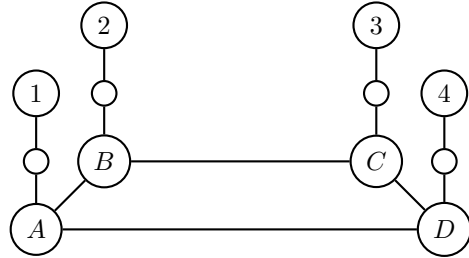
2.2 Fundamental Constructions

Throughout this paper we will repeatedly use two distinct and very useful constructions, for which we discuss their underlying idea in this section. The first one is our circling gadget and shows that, although existence of locally stable matchings is guaranteed for the bipartite case, there exist states for which REACHABILITY is not necessarily true. The second one is used to reduce 3SAT to finding a sequence of local improvement steps which traverse the gadget in a special order.

Circling Gadget The player set V consists of the classes $U = \{A, B, C, D, b1, b2, b3, b4\}$ and $W = \{1, 2, 3, 4\}$. The social links are $L = \{\{A, b1\}, \{B, b2\}, \{C, b3\}, \{D, b4\}, \{b1, 1\}, \{b2, 2\}, \{b3, 3\}, \{b4, 4\}, \{A, B\}, \{B, C\}, \{C, D\}, \{D, A\}\}$ (see picture below). For simplicity, we restrict the possible matching edges to $E = \{\{u, v\} \mid u = 1, \dots, 4, v = A, \dots, D\}$. It is obvious that by ranking $b1, \dots, b4$ at the bottom, we can similarly allow $E = U \times W$ without changing the argumentation.

The preference-lists are given by:

v	preferences
1	$C \succ B \succ A \succ D$
2	$D \succ C \succ B \succ A$
3	$A \succ D \succ C \succ B$
4	$B \succ A \succ D \succ C$
A	$4 \succ 1 \succ 3 \succ 2$
B	$1 \succ 2 \succ 4 \succ 3$
C	$2 \succ 3 \succ 1 \succ 4$
D	$3 \succ 4 \succ 2 \succ 1$

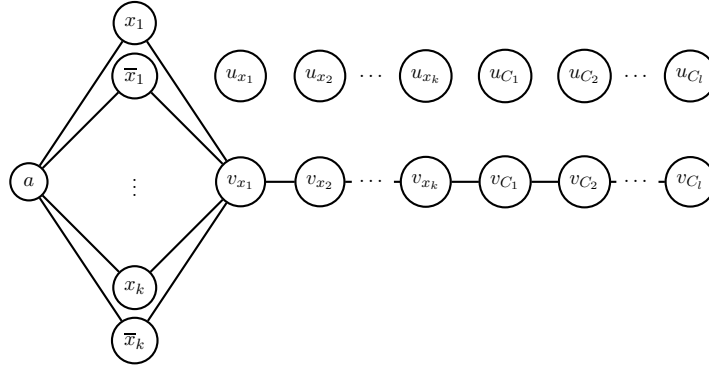


Dynamics: This gadget has two locally stable matchings, namely $\{\{1, B\}, \{2, C\}, \{3, D\}, \{4, A\}\}$ and $\{\{1, C\}, \{2, D\}, \{3, A\}, \{4, B\}\}$. However, from every state in which one vertex of W is unmatched, every possible sequence of local improvement steps leads to some state where some vertex of W is unmatched again. Because of symmetry, w.l.o.g. we can assume that 1 is unmatched. If A is not matched to 4, 1 can match to A and from there to B which means, when $\{B, 1\}$ is created

some other vertex of $\{1, \dots, 4\}$ is unmatched. If A is matched to 4 and B is not matched to 2, 4 can switch to B which frees A for 1. Otherwise, 2 can move to C (as it is C 's favourite partner) and then 4 can switch to B . Hence, every sequence of moves yields a state in which another vertex of $\{1, \dots, 4\}$ is unmatched. Then, this vertex takes the role of 1 and we can repeat the argument. In turn, it is simple to verify that when player 1 is matched to some more preferred partner outside of the gadget, the remaining players can always stabilize easily through local improvement steps.

Below we will use this gadget if we need to force a certain vertex to be matched in a locally stable matching. For this we will identify the forced vertex with 1 of our gadget and declare all allowed outside connections preferable to the gadget vertices. Then, if the vertex is matched to some vertex outside the gadget, the gadget cannot circle anymore while otherwise it will not stabilize.

3SAT Gadget We will use this gadget in different proofs of NP-hardness to implement the reduction from 3SAT. If the given 3SAT-formula contains k variables x_1, \dots, x_k and l clauses C_1, \dots, C_l , where clause C_j holds the literals l_{1j}, l_{2j} and l_{3j} , our gadget consists of a vertex u_{x_i} in U and x_i, \bar{x}_i and v_{x_i} in W for every variable x_i , as well as vertices u_{C_j} in U and v_{C_j} in W for every clause C_j . Further we have a vertex a in W that below will be replaced by other structures depending on which problem we want to reduce 3SAT to. For the social links see the picture shown below.



For simplicity, we restrict the allowed matching edges to

$$\begin{aligned}
E = & \quad \{\{u_{x_i}, a\}, \{u_{x_i}, x_i\}, \{u_{x_i}, \bar{x}_i\} \mid i = 1, \dots, k\} \\
& \cup \quad \{\{u_{C_j}, a\}, \{u_{C_j}, l_{1j}\}, \{u_{C_j}, l_{2j}\}, \{u_{C_j}, l_{3j}\} \mid j = 1, \dots, l\} \\
& \cup \quad \{\{u_{x_i}, v_{x_{i'}}\} \mid i = 1, \dots, k, i' = 1, \dots, i\} \\
& \cup \quad \{\{u_{C_j}, v_{x_i}\}, \{u_{C_j}, v_{C_{j'}}\} \mid i = 1, \dots, k, j = 1, \dots, l, j' = 1, \dots, j\} .
\end{aligned}$$

Again, by suitable definition of preferences we can allow all matching edges and achieve the same key properties as detailed in our discussion below.

Dynamics: Consider a u -vertex with a matching edge to a . The preferences and local improvements are such that u iteratively prefers to move the edge from a through the branching and then along the path up to its associated v -vertex. However, we will make sure by suitable constructions below that in each case a is reached by edges to the variable u -vertices before the edges to clause u -vertices. On the other hand, the v -vertices for the clauses lie further down along the path. To reach a matching in which every u -vertex is matched to the corresponding v -vertex, the clause-edges have to overtake the variable-edges at some point. This is only possible within the branching. By the allowed matching edges (or, more generally, suitable preferences) we ensure that for each variable u -vertex, it represents an improvement only if the edge is moved from a to one of the two corresponding variable vertices within the branching. For each clause u -vertex, it is an

improvement only if the edge is moved from a to vertices corresponding to literals appearing in the clause. Thus, if the 3SAT formula was satisfiable, each edge to a variable u -vertex can be parked in the branching at the inverse of its value in the satisfying assignment. As in every clause at least one literal is satisfied, this leaves a free path through the branching for every clause to bypass the variables. On the other hand, if the edges to variable u -vertices can be parked within the branching such that all clause-edges can bypass them, then assigning to each variable the value left open by its edge provides a satisfying assignment.

3 Reachability in Bipartite Network Games

In this section we focus on lower bounds for the REACHABILITY problem in general network games. Throughout, we focus on the empty matching as the initial matching and show that REACHABILITY is NP-hard to decide. This is in contrast to correlated network games, where REACHABILITY is true and for every initial matching there is always a polynomial sequence to a locally stable matching [8]. However, we here show that given a distinct matching to reach, deciding REACHABILITY becomes NP-hard, even for correlated job-market games.

Additionally, we give a network game and an initial matching so that we need an exponential number of steps before reaching any locally stable matching. This is again in contrast to the correlated case, where every reachable stable matching can be reached by a polynomial number of improvement steps.

3.1 Complexity

Theorem 1. *It is NP-hard to decide REACHABILITY from the initial matching $M = \emptyset$ to a given locally stable matching in a correlated bipartite network game.*

Proof. We use a reduction from 3SAT based on the gadget introduced above. Given a 3SAT formula with k variables x_1, \dots, x_k and l clauses C_1, \dots, C_l , where clause C_j contains the literals $l1_j, l2_j$ and $l3_j$, we amend the 3SAT gadget by vertices b_h , $h = 1, \dots, l + k - 1$, in U and a vertex a_1 in W . Further we add social links $\{a, a_1\}$, $\{a_1, u_{C_1}\}$, $\{u_{C_j}, b_j\}$ for $j = 1, \dots, l$, $\{b_j, u_{C_{j+1}}\}$ for $j = 1, \dots, l - 1$, $\{b_l, u_{x_1}\}$, $\{u_{x_i}, b_{l+i}\}$ for $i = 1, \dots, k - 1$ and $\{b_{l+i}, u_{x_{i+1}}\}$ for $i = 1, \dots, k - 1$. The set E is extended by $\{b_h, a\}$, $h = 1, \dots, l + k - 1$.

The benefits are given as follows.

$u \in U$	$w \in W$	edge weight $b(\{u, w\})$	
u_{C_j}	a	j	$j = 1, \dots, l$
u_{x_i}	a	$i + l$	$i = 1, \dots, k$
b_h	a	$h + \frac{1}{2}$	$h = 1, \dots, l + k - 1$
u_{C_j}	$l1_j/l2_j/l3_j$	$k + l + 1$	$j = 1, \dots, l$
u_{x_i}	x_i/\bar{x}_i	$k + l + 1$	$i = 1, \dots, k$
u_{C_j}	v_{x_i}	$k + l + 1 + i$	$i = 1, \dots, k, j = 1, \dots, l$
u_{x_i}	$v_{x_{i'}}$	$k + l + 1 + i'$	$i = 1, \dots, k, i' = 1, \dots, i$
u_{C_j}	$v_{C_{j'}}$	$2k + l + 1 + j'$	$j = 1, \dots, l, j' = 1, \dots, j$

Our goal is to reach $M^* = \{\{u_s, v_s\} | s \in \{x_1, \dots, x_k\} \cup \{C_1, \dots, C_l\}\}$.

We need to make sure that the edges reach a in the right order, then we can use the dynamics of the gadget as explained above. First, note that additional matching edges can only be introduced at $\{u_{C_1}, a\}$. Furthermore, once a vertex u_y , $y \in \{x_1, \dots, x_k\} \cup \{C_1, \dots, C_l\}$, is matched to a vertex other than a , it blocks the introduction of any edge for a vertex lying behind u_y on the path from

u_{C_1} to u_{x_k} . Also, the vertices b_h prevent that an edge is moved on from one u -vertex to another after it has left a . Thus, at the time when an edge to a clause u -vertex is created that still exists in the final matching (but is connected to some v_{C_j} then), the edges for all variable u -vertices must have been created already.

First assume that the 3SAT formula is satisfiable. Then we first create a matching edge at $\{u_{C_j}, a\}$, move it over the u - and b -vertices to u_{x_k} , and then move it into the branching to the one of x_k or \bar{x}_k that negates its value in the satisfying assignment. Similarly, one after the other (in descending order), we create matching edge at a for each of the variable u -vertices and move it into the branching to the variable vertex that negates its value in the satisfying assignment. As every clause is fulfilled, at least one of the three vertices that yield an improvement for the clause u -vertex from a is not blocked by a matching edge to a variable u -vertex. Then, the edges to clause u -vertices can bypass the existing edges (again, one after the other in descending order) and reach their positions in M^* . After that, the variable-edges can leave the branching and move to their final position in the same order as before.

Now assume that we can reach M^* from \emptyset . We note that the edges to clause u -vertices have to overtake the edges to variable u -vertices somewhere on the way to reach their final position. The only place to do so is in the branching leading over the x_i and \bar{x}_i . Thus all variable-edges have to wait at some x_i or \bar{x}_i until the clause-edges have passed. But from a vertex u_{x_i} is only willing to switch to x_i or \bar{x}_i . Thus, every vertex blocks out a different variable (either in its true or in its false value). Similarly, a vertex u_{C_j} will only move further from a if it can reach one of its literals. Hence, if all clauses can bypass the variables, then for every clause there was one of its literals left open for passage. Thus, if we set each variable to the value that yields the passage for clause-edges in the branching, we obtain a satisfying assignment. \square

Corollary 1. *It is NP-hard to decide REACHABILITY to a given locally stable matching in a correlated job-market game.*

Proof. We modify the proof of Theorem 1. We discard the vertices a_1 and all b_h as well as all incident links and matching-edges. This leaves every u_s , $s \in \{x_1, \dots, x_k, C_1, \dots, C_l\}$, isolated w.r.t. social links. Next we introduce new vertices a_s for $s \in \{x_1, \dots, x_k, C_1, \dots, C_l\}$ to W and links $\{a, a_{x_k}\}$, $\{a_{x_i}, a_{x_{i-1}}\}$ for $i = 2, \dots, k$, $\{a_{x_1}, a_{C_l}\}$ and $\{a_{C_j}, a_{C_{j-1}}\}$ for $j = 2, \dots, l$. As additional potential matching edges we have $\{\{u_s, a_{s'}\} | s, s' \in \{x_1, \dots, x_k, C_1, \dots, C_l\}\}$. The benefits are given by:

$u \in U$	$w \in W$	edge weight $b(\{u, w\})$	
u_s	a_{C_j}	j	$j = 1, \dots, l, s \in \{x_1, \dots, x_k, C_1, \dots, C_l\}$
u_s	a_{x_i}	$l + i$	$i = 1, \dots, k, s \in \{x_1, \dots, x_k, C_1, \dots, C_l\}$
u_s	a	$l + k + 1$	$s \in \{x_1, \dots, x_k, C_1, \dots, C_l\}$
u_{C_j}	$l1_j/l2_j/l3_j$	$k + l + 2$	$i = 1, \dots, k$
u_{x_i}	x_i/\bar{x}_i	$k + l + 2$	$i = 1, \dots, k$
u_{C_j}	v_{x_i}	$k + l + 2 + i$	$i = 1, \dots, k$
u_{x_i}	$v_{x_{i'}}$	$k + l + 2 + i'$	$i = 1, \dots, k, i' = 1, \dots, i$
u_{C_j}	$v_{C_{j'}}$	$2k + l + 2 + j'$	$j = 1, \dots, l, j' = 1, \dots, j$

Now if we start from $M_0 = \{\{u_y, a_y\} | y \in \{x_1, \dots, x_k, C_1, \dots, C_l\}\}$ the clause-edges have to overtake the variable-edges in the same manner as in Theorem 1 to reach $M^* = \{\{u_y, v_y\} | y \in \{x_1, \dots, x_k, C_1, \dots, C_l\}\}$. Thus our argumentation from above can be used to prove the result. \square

Theorem 2. *It is NP-hard to decide REACHABILITY from the initial matching $M = \emptyset$ to an arbitrary locally stable matching in a bipartite network game.*

Proof. Again we will reduce 3SAT to our problem. We use the same structure as in the proof of Theorem 1 and now make use of circling gadgets. For every variable and every clause we use one circling gadget and identify their v -vertex with the 1-vertex of the circling gadget. The preferences in the main structure are set as follows:

v	preferences	
u_{x_i}	$v_{x_i} \succ g_{x_{i-1}} \succ \dots \succ v_{x_1} \succ x_i \succ \bar{x}_i \succ a$	$i = 1, \dots, k$
u_{C_j}	$v_{C_j} \succ \dots \succ v_{C_1} \succ v_{x_k} \succ \dots \succ v_{x_1} \succ l1_j \succ l2_j \succ l3_j \succ a$	$j = 1, \dots, l$
a	$u_{x_k} \succ u_{x_{k-1}} \succ \dots \succ u_{x_1} \succ u_{C_l} \succ u_{C_{l-1}} \succ \dots \succ u_{C_1}$	
v_{x_i}	$u_{C_l} \succ \dots \succ u_{C_{j+1}} \succ u_{C_j} \succ 1_{x_i} \succ C_{C_j} \succ B_{C_j} \succ A_{C_j} \succ D_{C_j}$	$j = 1, \dots, l$
x_i/\bar{x}_i	$u_{x_i} \succ u_{C_1} \succ u_{C_2} \succ \dots \succ u_{C_l}$	$i = 1, \dots, k$

We only have to reason why we are forced to generate the edges $\{\{u_s, e_s\} \mid s \in \{x_1, \dots, x_k, C_1, \dots, C_l\}\}$ for every reachable locally stable matching, then we can use the argumentation of Theorem 1 to show the correctness of our reduction. Assume there exists a reachable locally stable matching with some v_s not matched to any of the u -vertices. For this s the circling gadget will not stabilize. Now, of all u_s , v_{C_l} can only be matched to u_{C_l} , which leaves only $u_{C_{l-1}}$ for $v_{C_{l-1}}$. Inductively repeating this argument results in all the edges $\{\{u_s, e_s\} \mid s \in \{x_1, \dots, x_k, C_1, \dots, C_l\}\}$ being necessary for every reachable locally stable matching. \square

3.2 Length of Sequences

Theorem 3. *For every network game with correlated preferences, every locally stable matching $M^* \in E$ and initial matching $M_0 \in E$ such that M^* can be reached from M_0 through local improvement steps, there exists a sequence of at most $O(|E|^3)$ local improvement steps leading from M_0 to M^* .*

Proof. Consider an arbitrary sequence between M_0 and M^* . We will explore which steps in the sequence are necessary and which parts can be omitted. We rank all edges by their benefit (allowing multiple edges to have the same rank) such that $r(e) > r(e')$ iff $b(e) > b(e')$ and set $r_{max} = \max\{r(e) \mid e \in E\}$. Recall from Section 2.1 that we can account edges in the way that every edge e has at most one direct predecessor e' in the sequence, which was necessary to build e . Because e was a local blocking pair, we know $r(e') < r(e)$. Thus, every edge e has at most r_{max} predecessors. Our proof is based on two crucial observations:

- (1) An edge can only be deleted by a stronger edge, that is, every chain of one edge deleting the next is limited in length by r_{max} .
- (2) If an edge is created, then possibly moved, and finally deleted without deleting an edge on its way, this edge would not have to be introduced in the first place.

Suppose our initial matching is the empty matching, then every edge in the locally stable matching has to be created and by (repeated application of) (2) we only need to create and move edges that are needed for the final matching. Thus we have $|M^*|$ edges, which each made at most r_{max} steps.

Now if we start with an arbitrary matching, the sequence might be forced to delete some edges that cannot be used for the final matching. Each of these edges generates a chain of edges deleting each other throughout the sequence, but (1) tells us that this chain is limited as well as the number of steps each of these edges has to make. The only remaining issue is what happens to edges "accidentally" deleted during this procedure. Again, we can use (2) to argue that there is no reason to rebuild such an edge just to delete it again. Thus, such deletions can happen only once for every edge we had in M_0 (not necessarily on the position it had in M_0). It does not do any harm if it happens to an edge of one of the deletion-chains, as it would just end as desired. For the

edges remaining in $|M^*|$ the same bounds holds as before. Thus, we have an overall bound of $|M_0| \cdot r_{\max} \cdot r_{\max} + |M^*| \cdot r_{\max} \in O(|E|^3)$ steps, where the first term results from the deletion chains and the second one from the edges surviving in the final matching. \square

Theorem 4. *There is a bipartite network game with general preferences such that a locally stable matching can be reached by a sequence of local improvement steps from the initial matching $M = \emptyset$, but every such sequence has length $2^{\Omega(|V|)}$.*

Proof. We will construct a network consisting of $n = \Omega(|V|)$ entangled gadgets $1, \dots, n$ of constant size. The intuition of our construction is as follows. We first ensure that in every gadget exactly two edges are created that can rotate inside this gadget. Then, we force the dynamics to move edges through the network to specific positions that are occupied in every reachable stable matching. The idea of our construction is that edges need to pass through other gadgets. In particular, for an edge that enters a gadget from outside and wants to pass through, both edges inside the gadget have to move out of the way. Additional outside-edges can only pass a gadget one by one and hence cannot take advantage of an inside-edge moved out of the way for some previous outside-edge. Now, by the way the gadgets are intertwined, to get out of the way, an edge of gadget i has to pass through gadget $i + 1$. Thus, an edge that has to bypass gadget 1 forces both edges of this gadget to move, resulting in each of them forcing both edges of gadget 2 to move and so on. Finally, edges in gadget n are forced to be moved 2^n times.

We now describe our construction more formally. We will build the network from a number of gadgets that have different functions. For clarity we list and analyze them separately as far as possible.

Generation and activation gadget:

Vertices:

$A_0, \text{Distribute}_0, \text{End}_0$

Links:

$\{\{A_0, C_i\}, \{\text{Distribute}_0, D_i\} | i = 1, \dots, n\},$
 $\{F_1, \text{End}_0\}$

Matching-edges:

$\{\{A_0, D_i\} | i = 1, \dots, n\}, \{A_0, 1_1\}, \{A_0, E_1\},$
 $\{A_0, 2_1\}, \{A_0, F_1\}, \{A_0, \text{End}_0\}$

v	preferences
A_0	$\text{End}_0 \succ F_1 \succ 2_1 \succ E_1$ $\succ 1_1 \succ D_1 \succ D_2 \succ \dots \succ D_n$
Distribute_0	—
End_0	A_0

Rotating gadget $i = 1$:

Vertices:

$A_i, B_i, C_i, D_i, E_i, F_i, 1_i, 2_i, \text{End}1_i, \text{End}2_i$

Links:

$\{A_i, B_i\}, \{D_i, E_i\}, \{E_i, F_i\}, \{F_i, D_i\}, \{D_i, 1_i\},$
 $\{1_i, E_i\}, \{E_i, 2_i\}, \{2_i, F_i\}, \{A_i, \text{End}1_i\}, \{C_i, \text{End}2_i\}$

Matching-edges:

$\{A_i, E_i\}, \{A_i, F_i\}, \{B_i, D_i\}, \{B_i, E_i\},$
 $\{C_i, F_i\}, \{C_i, D_i\}, \{F_i, \text{End}1_i\}, \{D_i, \text{End}2_i\}$

v	preferences
A_i	$F_i \succ E_i$
B_i	$E_i \succ D_i$
C_i	$D_i \succ F_i$
D_i	$\text{End}2_i \succ B_i \succ F_{i+1} \succ 2_{i+1} \succ E_{i+1}$ $\succ 1_{i+1} \succ D_{i+1} \succ C_i \succ A_0$
E_i	$A_i \succ B_i \succ A_0$
F_i	$\text{End}1_i \succ C_i \succ F_{i+1} \succ 2_{i+1} \succ E_{i+1}$ $\succ 1_{i+1} \succ D_{i+1} \succ A_i \succ A_0$
1_i	A_0
2_i	A_0
$\text{End}1_i$	F_i
$\text{End}2_i$	D_i

Rotating gadget $i = 2, \dots, n-1$:

Vertices:

$A_i, B_i, C_i, D_i, E_i, F_i, 1_i, 2_i, End1_i, End2_i$

Links:

$\{A_i, B_i\}, \{D_i, E_i\}, \{E_i, F_i\}, \{F_i, D_i\},$
 $\{D_i, 1_i\}, \{1_i, E_i\}, \{E_i, 2_i\}, \{2_i, F_i\},$
 $\{A_i, End1_i\}, \{C_i, End2_i\}, \{D_i, A_{i-1}\},$
 $\{D_i, C_{i-1}\}, \{F_i, B_{i-1}\}, \{F_i, C_{i-1}\}$

Matching-edges:

$\{A_i, E_i\}, \{A_i, F_i\}, \{B_i, D_i\}, \{B_i, E_i\},$
 $\{C_i, F_i\}, \{C_i, D_i\}, \{F_i, End1_i\}, \{D_i, End2_i\},$
 $\{D_i, D_{i-1}\}, \{1_i, D_{i-1}\}, \{E_i, D_{i-1}\}, \{2_i, D_{i-1}\},$
 $\{F_i, D_{i-1}\}, \{D_i, F_{i-1}\}, \{1_i, F_{i-1}\},$
 $\{E_i, F_{i-1}\}, \{2_i, F_{i-1}\}, \{F_i, F_{i-1}\}$

v	preferences
A_i	$F_i \succ E_i$
B_i	$E_i \succ D_i$
C_i	$D_i \succ F_i$
D_i	$End2_i \succ B_i \succ F_{i+1} \succ 2_{i+1} \succ E_{i+1} \succ 1_{i+1} \succ D_{i+1} \succ C_i \succ D_{i-1} \succ F_{i-1} \succ A_0$
E_i	$A_i \succ B_i \succ D_{i-1} \succ F_{i-1}$
F_i	$End1_i \succ C_i \succ F_{i+1} \succ 2_{i+1} \succ E_{i+1} \succ 1_{i+1} \succ D_{i+1} \succ A_i \succ D_{i-1} \succ F_{i-1}$
1_i	$D_{i-1} \succ F_{i-1}$
2_i	$D_{i-1} \succ F_{i-1}$
$End1_i$	F_i
$End2_i$	D_i

Rotating gadget $i = n$:

Vertices:

$A_i, B_i, C_i, D_i, E_i, F_i, 1_i, 2_i, End1_i, End2_i$

Links:

$\{A_i, B_i\}, \{A_i, C_i\}, \{B_i, C_i\}, \{D_i, E_i\},$
 $\{E_i, F_i\}, \{F_i, D_i\}, \{D_i, 1_i\}, \{1_i, E_i\},$
 $\{E_i, 2_i\}, \{2_i, F_i\}, \{A_i, End1_i\}, \{C_i, End2_i\},$
 $\{D_i, A_{i-1}\}, \{D_i, C_{i-1}\}, \{F_i, B_{i-1}\}, \{F_i, C_{i-1}\}$

Matching-edges:

$\{A_i, E_i\}, \{A_i, F_i\}, \{B_i, D_i\}, \{B_i, E_i\},$
 $\{C_i, F_i\}, \{C_i, D_i\}, \{F_i, End1_i\}, \{D_i, End2_i\},$
 $\{D_i, D_{i-1}\}, \{1_i, D_{i-1}\}, \{E_i, D_{i-1}\}, \{2_i, D_{i-1}\},$
 $\{F_i, D_{i-1}\}, \{D_i, F_{i-1}\}, \{1_i, F_{i-1}\},$
 $\{E_i, F_{i-1}\}, \{2_i, F_{i-1}\}, \{F_i, F_{i-1}\}$

v	preferences
A_i	$F_i \succ E_i$
B_i	$E_i \succ D_i$
C_i	$D_i \succ F_i$
D_i	$End2_i \succ B_i \succ C_i \succ D_{i-1} \succ F_{i-1} \succ A_0$
E_i	$A_i \succ B_i \succ D_{i-1} \succ F_{i-1}$
F_i	$End1_i \succ C_i \succ A_i \succ D_{i-1} \succ F_{i-1}$
1_i	$D_{i-1} \succ F_{i-1}$
2_i	$D_{i-1} \succ F_{i-1}$
$End1_i$	F_i
$End2_i$	D_i

The bipartite partition of the vertex set is as follows:

$$\begin{aligned}
 U &= \{A_0\} \cup \{A_i, B_i, C_i, End1_i, End2_i \mid i \text{ odd}\} \cup \{D_i, 1_i, E_i, 2_i, F_i \mid i \text{ even}\}, \\
 W &= \{Distribute_0, End_0\} \cup \{A_i, B_i, C_i, End1_i, End2_i \mid i \text{ even}\} \cup \{D_i, 1_i, E_i, 2_i, F_i \mid i \text{ odd}\}.
 \end{aligned}$$

The intertwining of gadgets is achieved by letting every *End*-vertex be the 1-vertex of a circling gadget. It ranks the vertices inside the circling gadget lowest.

The only way to reach a locally stable matching is to block each of the circling gadgets via an edge between the *End*-vertex and the vertex in its associated rotating gadget respectively in the activation gadget. Hence, a reachable stable matching holds at least two edges in every rotating gadget as well as the edge $\{A_0, End_0\}$.

Now, the only way to generate matching edges for the rotating gadget i is at $\{A_0, D_i\}$. This has to be done before the edge ending at $\{A_0, End_0\}$ is created and sent on its way, as it blocks the vertex A_0 throughout its entire journey through the network. We term this edge the activation edge, and we observe that none of the edges needed for the final matching should be deleted once the activation edge is created, because it can not be re-created. Furthermore, every rotating gadget can hold at most two edges without being blocked, as every edge uses one of the vertices D_i, E_i, F_i

and one of these players has to be unmatched to allow for movement. Movement within gadgets is necessary, as the activation edge moves along the first rotating gadget, forcing its edges to move out of the way, which again results in the edges of the second gadget having to move, and so on. Thus every rotating gadget holds exactly 2 edges in the final matching, namely $\{F_i, \text{End1}_i\}$ and $\{D_i, \text{End2}_i\}$. Edges that got deleted by another edge passing do not contribute to the result in any way, so a shortest sequence will only generate the two edges for every rotating gadget, and afterwards the edge for the activation gadget as well as the edges for the circling gadgets and no more.

Once every rotating gadget is filled with two edges, the following invariant holds: Let e_1 be the first edge of gadget i according to the order $F_i \in e < E_i \in e < D_i \in e$ and let e_2 be the second one. To allow an edge to pass along the gadget, the vertices D_i, E_i, F_i have to become unmatched in this order. That is, at first e_1 is incident to F_i and e_2 to E_i , then e_1 to D_i and e_2 to F_i and finally e_1 to E_i and e_2 to D_i . Thus, e_1 needs to move from $\{C_i, D_i\}$ to $\{B_i, D_i\}$ and e_2 from $\{A_i, F_i\}$ to $\{C_i, F_i\}$. Both ways bypass gadget $i + 1$ (if $i < n$). This results in the activation edge forcing both edges of rotating gadget 1 to bypass gadget 2, which again results in the edges of 2 both moving along gadget 3 two times and so on, until in gadget n the edges need $\Omega(2^n)$ steps to (repeatedly) get out of the way. \square

4 Memory

We now focus on the impact of memory for the reachability of locally stable matchings. As a direct initial result, we observe that no memory can help with the reachability of a *given* locally stable matching, even in a correlated job-market game.

Corollary 2. *It is NP-hard to decide REACHABILITY to a given locally stable matching in a correlated job-market game with any kind of memory.*

Proof. We observe that the same reduction as in Corollary 1 yields the result even for arbitrary memory. If the 3SAT formula is satisfiable, we obviously need no memory to reach the desired locally stable matching. Now assume that the desired matching M^* is reachable. The crucial point is that memory can only yield accessible edges for u_y up to the point of the path that was already discovered by u_y . Furthermore, an edge belonging to a variable u -vertex can only be removed by a variable u -vertex of higher index. The first vertex along the path where an edge can be deleted is v_{x_1} . Thus, if we cannot store all variable-edges inside the branching to let all clause-edges pass, we have to push at least one variable-edge out of the branching before the last clause-edge reaches v_{x_1} . Then, however, there always will be a variable-edge in front of the last clause-edge, as the variable edges can only be deleted through other variable edges. In addition, the last clause-edge cannot use memory to jump ahead along the path. Hence, to reach M^* all variable-edges must be stored within the branching, while leaving at least one path open for every clause-edge. This implies the existence of a satisfying assignment as before. \square

We will now concentrate on the impact of memory on reaching an arbitrary locally stable matching. Although existence is guaranteed for bipartite matchings, even quite simple graphs like the circling gadget do not allow to reach a locally stable matching through improvement step dynamics. For our treatment we will focus on the case in which the network links $L \subseteq (W \times W) \cup (U \times W)$.

Quality Memory We start our treatment with quality memory, where each player remembers each round the best matching partner he ever had before. While this seems quite a natural choice

and appears like a smart strategy for each player, it can be easily fooled by starting with a much-liked partner, who soon after matches with someone more preferred and never becomes available again. This way the memory becomes useless which leaves us with the same dynamics as before.

Proposition 1. *There is a bipartite network game with general preferences, links $L \subseteq (W \times W) \cup (U \times W)$, quality memory and initial matching $M = \emptyset$ such that no locally stable matching can be reached with local improvement steps from M .*

Proof. The idea is to use the circling gadget and add a memory reset for each vertex. This additional construction is to make sure that in the beginning each of the ordinary gadget vertices gets to match with a vertex $u2$ it likes better than all the ones inside the gadget. Then, each of these $u2$ -vertices match to some other partner $u3$ but stay in the memory of gadget vertices indefinitely, thereby making their memory useless. Formally, we consider $U \cup \{u3_A, u3_B, u3_C, u3_D, u1_1, u2_1, u1_2, u2_2, u1_3, u2_3, u1_4, u2_4\}$ and $W \cup \{u3_1, u3_2, u3_3, u3_4, u1_A, u2_A, u1_B, u2_B, u1_C, u2_C, u1_D, u2_D\}$ as well as links $L \cup \{\{w, u1_w\}, \{u1_w, u2_w\}, \{w, u3_w\} \mid w = 1, \dots, 4, A, \dots, D\}$ and allowed matching-edges $E \cup \{\{w, u2_w\}, \{u2_w, u3_w\} \mid w = 1, \dots, 4, A, \dots, D\}$.

We start with $M = \emptyset$ and assume that there is a sequence of local improvement steps leading to a stable matching. Now a matching cannot be stable until the edge $\{w, u2_w\}$ was created once for all $w = 1, \dots, 4, A, \dots, D$, because otherwise (1) it is available for creation, (2) matches w to its favorite partner, and (3) matches $u2_w$ to the only possible partner it knows before being matched to w . Afterward, every such edge will move on to the position $\{u2_w, u3_w\}$ and stay there as it matches both vertices to their most preferred choice. Let w be the last vertex for which $\{u2_w, u3_w\}$ is generated. At that moment w is unmatched and every vertex of $\{1, \dots, 4, A, \dots, D\}$ will continue to hold its $u2$ -partner in its memory. As the $u2$ -vertices are not willing to change their matching edges, there will be no edge created from memory from this point on. If $w \in \{1, \dots, 4\}$, one of the vertices in $\{A, \dots, D\}$ is unmatched as well and vice versa. This leaves us in the situation described in the dynamics of the circling gadget with useless memory, and hence from this point on no locally stable matching can be reached. \square

Remark. The same memory reset also works if every player remembers the best k previous matches, for any number k , simply by applying k copies of the memory reset construction for each player.

Theorem 5. *It is NP-hard to decide REACHABILITY to an arbitrary locally stable matching in a bipartite network game with quality memory.*

Proof. The proof uses exactly the same argument as used in Theorem 2. We simply combine the 3SAT gadget described in Corollary 2 with our gadget from Proposition 1 in the same way as we did in Theorem 2. Then, for the 3SAT gadget the memory is useless. The circling gadget in Proposition 1 allows to reach a locally stable matching only if all memory resets have been executed. In this case, however, we need to reach a specific matching in the 3SAT construction. \square

Recency Memory In recency memory, every player remembers the last player he has been matched to before. This is as well a very natural choice as it expresses the human character of remembering the latest events best. Interestingly, in this case we actually can ensure reaching a locally stable matching under our constraints on the network structure.

Theorem 6. *For every bipartite network game with general preferences, links $L \subseteq (U \times W) \cup (W \times W)$, recency memory and every initial matching, there is a sequence of $O(|U|^2|W|^2)$ many local improvement steps to a locally stable matching.*

Proof. Our basic approach is to construct the sequence in two phases similarly as in [1]. In the first phase, we let the matched vertices from U improve, but ignore the unmatched ones. In the second phase, we make sure that vertices from W have improved after every round.

Preparation phase: As long as there is at least one $u \in U$ with u matched and u part of a blocking pair, allow u to switch to the better partner.

The preparation phase terminates after at most $|U| \cdot |W|$ steps, as in every round one matched $u \in U$ strictly improves in terms of preference. This can happen at most $|W|$ times for each matched u . In addition, the number of matched vertices from U only decreases.

Memory phase: As long as there is a $u \in U$ with u part of a blocking pair, pick u and execute a sequence of local improvement steps involving u until u is not part of any blocking pair anymore. For every edge $e = \{u', w\}$ with $u' \neq u$ that was deleted during the sequence, recreate e from the memory of u' .

We claim that if we start the memory phase after the preparation phase, at the end of every round we have the following invariants: The vertices from W that have been matched before are still matched, they do not have a worse partner than before, and at least one of them is matched strictly better than before. Also, only unmatched vertices from U are involved in local blocking pairs.

Obviously, at the end of the preparation phase the only U -vertices in local blocking pairs are unmatched, i.e., initially only unmatched U -vertices are part of local blocking pairs. Let u be the vertex chosen in the following round of the memory phase. At first we consider the outcome for $w \in W$. If w is the vertex matched to u in the end, then w clearly has improved. Otherwise w gets matched to its former partner (if it had one) through memory and thus has the same utility as before. In particular, every w that represents an improvement to some u' but was blocked by a higher ranked vertex still remains blocked. Together with the fact that u plays local improvement steps until it is not part of a local blocking pair anymore, this guarantees that all matched U -vertices cannot improve at the end of the round. As one W -vertex improves in every round, we have at most $|U| \cdot |W|$ rounds in the memory phase, where every round consists of at most $|W|$ steps by u and at most $|U| - 1$ edges reproduced from memory. \square

The following is a direct corollary from the previous theorem. In random dynamics, with probability 1 in the long run, we will at least once start in a matching and execute the sequence described in the last theorem. This yields the following corollary.

Corollary 3. *For every bipartite network game with general preferences, links $L \subseteq (U \times W) \cup (W \times W)$ recency memory, and every initial matching, random dynamics converge to a locally stable matching in the long run with probability 1.*

Sadly, we cannot expect fast convergence here, as there are instances where random dynamics yield an exponential sequence with high probability even if all information is given. In particular, we can take the instance from [1] and assume that the network contains all links $L = U \times W$. This means all players know all possible matching partners and memory has no effect.

Our constructions require only the players from U to have recency memory, which allows us to show the same results when we have no memory for vertices in W . In contrast, if we omit the memory for U and just keep the one for W , no sequence of local improvement steps might lead to a locally stable matching.

Proposition 2. *There is a bipartite network game with general preferences, links $L \subseteq (U \times W) \cup (W \times W)$, recency memory for vertices in W and initial matching $M = \emptyset$ such that no locally stable matching can be reached with local improvement steps from M .*

Proof. Consider the circling gadget using the given preference lists but slightly rearranging partitions such that $U = \{1, 2, 3, 4\}$ and $W = \{A, B, C, D, b1, b2, b3, b4\}$. Suppose only vertices in W have recency memory and that there is a sequence of local improvement steps which leads to a locally stable matching. First off note that the only edges, where memory can help us, are $\{A, 4\}$, $\{B, 1\}$, $\{C, 2\}$ and $\{D, 3\}$, as $\{A, 1\}$, $\{B, 2\}$, $\{C, 3\}$ and $\{D, 4\}$ are always known and the edges $\{A, 3\}$, $\{B, 4\}$, $\{C, 1\}$ and $\{D, 2\}$ only get deleted if the W -vertex finds a better partner – that is, he does not want to go back while matched with the new partner, and if the partner switches, it will overwrite its memory. As all vertices of W are matched in a stable matching and none of the vertices are in hop-distance 2 in L , the last edge to complete the matching must come from memory. This immediately rules out the matching $\{\{A, 3\}, \{B, 4\}, \{C, 1\}, \{D, 2\}\}$ and leaves us with the matching $M^* = \{\{A, 4\}, \{B, 1\}, \{C, 2\}, \{D, 3\}\}$. Now can we get a vertex to remember the relevant edge? Due to symmetry we only examine the situation for $\{A, 4\}$. We start our phase in a situation where A is matched to 4 (to get it into its memory) and no vertex can build an edge of M^* through memory (either because they do not remember it or because their partner is not available). If the edge $\{A, 4\}$ exists, A does not want to switch. Hence the edge gets lost, when 4 switches to B . Then 4 is happy, that is, $\{B, 4\}$ has to be destroyed, before A can use its memory to get 4 back. If $\{B, 2\}$ is built, we later on need to match 1 with B without the help of memory, that is, reset the cache of A on the way. If $\{B, 1\}$ is used, B must have remembered 1 to not destroy A 's memory of 4. Then 1 was not available at the beginning of the phase, that is, $\{C, 1\}$ existed and is now destroyed. This could only happen through C remembering 2 but 2 being matched to D in the beginning. This leads to D needing to remember 3 and 3 being matched to A , which contradicts $\{A, 4\}$. \square

The lemma can be used with the construction of Corollary 2 in the same manner as Proposition 1 was used to show Theorem 5. This yields the following result.

Theorem 7. *It is NP-hard to decide REACHABILITY to an arbitrary locally stable matching in a bipartite network game when recency memory exists only for one partition.*

Random Memory Let us now consider how random memory might help us reach a locally stable matching from every starting state even in general bipartite network games. Again, we cannot expect fast convergence due to the full-information lower bound in [1]. However, we show that random memory can help with reachability:

Theorem 8. *For every bipartite network game with random memory, random dynamics converge to a locally stable matching with probability 1.*

Proof. Our proof combines an idea used in [8] with a convergence result in [1]. We consider an infinite sequence of random local improvement steps and divide it into phases. Whenever a new edge is created for the first time, a new phase starts. Hence, phase t contains the part of the sequence, where exactly t edges have been created so far. Within a phase we have a fixed number of matching edges available (from the network or from memory). Consider phase t^* where t^* is the maximal phase of the sequence. t^* exists, as t is monotonically increasing and limited by $|E|$. We know that every phase $t < t^*$ ends after a finite number of steps. Hence we only have to show, that phase t^* is finite in expectation as well. The proof of Theorem 4 in [1] demonstrates how to construct a sequence of blocking pair resolutions to form a stable matching in the full information

case when all possible matching edges are known. In phase t^* we have t^* edges that can be used for the matching and all of them can also be remembered. Thus, with strictly positive probability there is an initial state such that the random memory remembers the blocking pairs from the sequence in the correct order and the random dynamics implement the blocking pair resolutions in the correct way. Thus, phase t^* is expected to end after a finite number of steps. This proves the theorem. \square

5 Centralized Problems

5.1 Maximum Locally Stable Matching

The size of locally stable matchings can vary significantly – up to the point where the empty matching as well as a matching that includes every vertex is locally stable – it is desirable in terms of social welfare to target locally stable matchings of maximal size. In this section we will address the computational complexity of finding maximum locally stable matchings. It turns out that there is a close connection between the maximum independent set problem and the maximum locally stable matching problem, which implicates that hardness of approximation results for independent set transfer to locally stable matching. We first use instances of maximum independent set to build hard instances for maximum locally stable matching.

Theorem 9. *For every graph G there is a job-market game that admits a maximum locally stable matching of size $|V[G]| + k$ if and only if G holds a maximum independent set of size k .*

Proof. Given a graph $G = (V, E)$, $|V| = n$, we construct the job-market game with network $N = (V' = U \cup W, L)$. For every $v \in V$ we have $u_{v,1}, u_{v,2} \in U$ and $w_{v,1}, w_{v,2} \in W$. We have the links $\{w_{v,1}, w_{v',2}\}$ and $\{w_{v',2}, w_{v,2}\}$ if $v' \in N(v)$. We allow matching edges $\{u_{v,1}, w_{v,1}\}$, $\{u_{v,1}, w_{v',2}\}$ for $v' \in N(v)$, $\{u_{v,1}, w_{v,2}\}$ and $\{u_{v,2}, w_{v,2}\}$. Each $u_{v,1}$ prefers $w_{v,2}$ to every $w_{v',2}$, $v' \in N(v)$, and every $w_{v',2}$ to $w_{v,1}$. The preferences between the different neighbors can be chosen arbitrary. Each $w_{v,2}$ prefers $u_{v,1}$ to every $u_{v',1}$, $v' \in N(v)$, and every $u_{v',2}$ to $u_{v,2}$. Again the neighbors can be ordered arbitrary. The vertices $w_{v,1}$ and $u_{v,2}$ have only one possible matching partner anyway.

We claim that G has a maximum independent set of size k iff N has a locally stable matching of size $n + k$.

Let S be a maximum independent set in G . Then $M = \{\{u_{v,1}, w_{v,2}\} \mid v \in V \setminus S\} \cup \{\{u_{v,1}, w_{v,1}\}, \{u_{v,2}, w_{v,2}\} \mid v \in S\}$ is a locally stable matching as the edges $\{u_{v,1}, w_{v,2}\}$ are always stable. For the other vertices the independent set property tells us that for $v \in S$ all vertices $v' \in N(S)$ generate stable edges $\{u_{v',1}, w_{v',2}\}$ that keep $u_{v,1}$ from switching to $w_{v',2}$. Thus $\{u_{v,1}, w_{v,1}\}$ is stable and $w_{v,2}$ cannot see $u_{v,1}$ which stabilizes $\{u_{v,2}, w_{v,2}\}$.

Now let M be a maximum locally stable matching for the job-market game. Further we chose M such that every $u_{v,1}$ is matched, which is possible as replacing a matching partner of $w_{v,2}$ by (the unmatched) $u_{v,1}$ will not generate instabilities or lower the size of M . We note that no $u_{v,1}$ is matched to some $w_{v',2}$ with $v \neq v'$ as from there $u_{v,1}$ and $w_{v,2}$ can see each other and thus constitute a blocking pair. Then for $S = \{v \mid u_{v,2} \in M\}$ $|S| = |M| - n$ and S is an independent set, as every $u_{v,2}$ can only be matched to its vertex $w_{v,2}$, which means that $u_{v,1}$ must be matched to $w_{v,1}$. But this edge is only stable if every $w_{v',2}$, $v' \in N(v)$, is blocked by $u_{v',1}$. Hence for every $v \in S$ $N(v) \cap S = \emptyset$. \square

Corollary 4. *Finding a maximum locally stable matching is NP-complete. Under the unique games conjecture the problem cannot be approximated within $1.5 - \varepsilon$, for any constant ε .*

Proof. We will use the relation to independent set in Theorem 9 and the result of [4] stating that independent set is unique games-hard to approximate within a factor of $O\left(\frac{d}{\log^2(d)}\right)$ for independent sets of size $k = \left(\frac{1}{2} - \Theta\left(\frac{\log(\log(d))}{\log(d)}\right)\right)n$ where n is the size of the vertex set and d is an upper bound on the degree. Setting $d = \delta n$ for some constant $\delta > 0$, maximum locally stable matching is unique games-hard to approximate within

$$\frac{n + \left(\frac{1}{2} - \Theta\left(\frac{\log(\log(n))}{\log(n)}\right)\right)n}{n + \left(\frac{1}{2} - \Theta\left(\frac{\log(\log(n))}{\log(n)}\right)\right)n \cdot O\left(\frac{\log(n)}{n}\right)} \geq 1.5 - \varepsilon,$$

for sufficiently large n . \square

In fact, our reduction applies in the setting of the bipartite job-market game, where one side has no network at all. This shows that even under quite strong restrictions the hardness of approximation holds. In contrast, it is easy to obtain a 2-approximation in every network game that admits a globally stable matching.

Proposition 3. *If a (globally) stable matching exists, every such stable matching is a 2-approximation for the maximum locally stable matching.*

Proof. Recall that every stable matching is locally stable as well. Now let M be a stable matching and $e = \{u, v\}$ an edge of a maximum locally stable matching M^* . We show that at least one vertex of e is matched in M . Then, obviously, $|M^*| < 2|M|$. Assume that both vertices are unmatched in M . As e exists in M^* , u and v prefer each other to being alone. Thus (u, v) is a blocking-pair and M cannot be stable. \square

The following theorem shows that a close relationship between locally stable matching and independent set holds in the reverse direction as well.

Theorem 10. *For every network game with n players there is a graph G with a maximum independent set of size $n^2 - n + k$ iff the game has a maximum locally stable matching of size k . If the game has no locally stable matching, G has a maximum independent set of size strictly less than $n^2 - n$.*

Proof. For convenience, we will show the result for maximum weighted independent set, i.e., we allow the vertices in G to have integer weights. However, we note that our construction can easily be translated to the unweighted independent set problem by introducing a number of copies for each vertex that correspond to its weight.

Given a network game with network $N = (V, L)$, $|V| = n$, and a set $E \subseteq V \times V \setminus \{\{v, v\} | v \in V\}$ as well as preference orders \succ_v , $v \in V$, let

$$B_{v,w} = \{w' | w' \neq w \in V, w' \text{ within distance of 2 from } v \text{ in } (V, L \cup \{e\}), w' \succ_v w\}$$

be the set of all vertices that v would prefer to w if matched with w . Similarly, we define

$$B_v = \{w | v \neq w \in V, w \text{ within distance of 2 from } v \text{ in } (V, L), \{v, w\} \in E\}$$

to be the set of all vertices that v prefers to being unmatched.

We construct a graph $G = (V', E')$ for the maximum independent set problem as follows. For every $v \in V$ we have a vertex v symbolizing that v is unmatched and for every edge $e \in E$ we have

a vertex e symbolizing that e is part of the matching. For the edges we have

$$\{\{e, e'\} | e, e' \in E, e \cap e' \neq \emptyset\} \cup \{\{v, e\} | e \in E, v \in e\}$$

to ensure that no vertex occurs twice and further

$$\begin{aligned} & \{\{\{v, w\}, \{v', w'\}\} | \{v, w\}, \{v', w'\} \in E, w' \in B_{v,w}, v \succ_{w'} v'\} \\ & \cup \{\{v, \{v', w'\}\} | v \in V, \{v', w'\} \in E, w' \in B_v, v \succ_{w'} v'\} \\ & \cup \{\{\{v, w\}, w'\} | w' \in V, \{v, w\} \in E, w' \in B_{v,w}\} \cup \{\{v, w\} | v, w \in V, w \in B_v\} \end{aligned}$$

to ensure that for a chosen edge or vertex, all edges and vertices that lead to an instable situation (blocking pair) are not in the independent set. The edge-vertices have weight $w_e = 2n - 1$ and the vertex-vertices have weight $w_v = n - 1$.

At first, we assume that our game has a maximum locally stable matching M of size k . Then the set

$$S = \{e | e \in M\} \cup \{v | v \in V \setminus \cup_{e \in M} e\}$$

is an independent set in G of size

$$\sum_{e \in M} w_e + \sum_{v \in V \setminus \cup_{e \in M} e} w_v = kw_e + (n - 2k)w_v = n^2 - n + k.$$

Now let S' be a maximum independent set in G . We have seen above that a maximum matching of size k in the game allows an independent set of weight $n^2 - n + k$ which is always at least $n^2 - n$. Thus, if S' has weight $< n^2 - n$, then the game cannot have a locally stable matching. Now let us consider the case that S' has weight $\geq n^2 - n$. We note that for every $v \in V$ we can have at most one vertex of $V_v = \{v\} \cup \{e | v \in e \in E\}$ in S' and if we have $V_v \cap S' = \emptyset$ for some v , the weight of S' is at most

$$\left\lfloor \frac{n-1}{2} \right\rfloor w_e + ((n-1) \bmod 2)w_v \leq \frac{n-1}{2}(2n-1) = n^2 - \frac{3n}{2} + \frac{1}{2} < n^2 - n.$$

Thus S' contains exactly one vertex of each V_v and has a weight of $\sum_{e \in S'} w_e + \sum_{v \in S'} w_v = n^2 - n + |\{e | e \in S'\}|$. Let $M' = \{e | e \in S'\}$. Then M' is a locally stable matching, as all vertices in S' are chosen such that their combination does not contain a blocking pair (as vertices representing a blocking pair are connected) and S' covers every V_v , that is, S' represents a combination involving all $v \in V$ and not allowing instable situations. Now, as S' was chosen to be a maximum independent set, M' must be a maximum locally stable matching, because we have seen above that a bigger locally stable matching would provide a bigger independent set as well. \square

5.2 Roommates Problem

Existence For bipartite network games there always exists a stable matching and thus a locally stable one as well (although it might not be reachable by local improvement steps, as seen above). Contrariwise in the roommates problem, in which all possible pairs can form, there are games such that no stable matching exists. The same obviously holds for locally stable matchings, as for a complete network N every locally stable matching must be globally stable as well. For the ordinary roommates problem, existence of a stable matching can be decided in polynomial time [9, 19]. In contrast, we here show that existence of a locally stable matching cannot be decided in polynomial

time.

Theorem 11. *It is NP-hard to decide existence of a locally stable matching in a network game with general preferences.*

Proof. We use a reduction from 3SAT, but this time our construction will not work with the 3SAT gadget presented above. For existence we cannot use the improvement step dynamics that forced the edges along certain branches of the gadget and thereby gave information about the satisfiability of our formula. Instead, we will use a circling structure for each clause which has to be blocked by edges representing the variable assignment. Our construction uses two copies of the same graph. This is to control the matching decisions of the vertices we do not need for the relevant part of the matching; these vertices are matched to their copy.

Given a 3SAT formula with k variables x_1, \dots, x_k and l clauses C_1, \dots, C_l , we will build two identical graphs and then connect every vertex with its copy. For every variable x_i we have vertices $x_i, \bar{x}_i, 1_{x_i}, 2_{x_i}, 3_{x_i}, 4_{x_i}, 5_{x_i}$ as well as vertices $x'_i, \bar{x}'_i, 1'_{x_i}, 2'_{x_i}, 3'_{x_i}, 4'_{x_i}, 5'_{x_i}$ for the copy. For every clause C_j we have vertices $A_{C_j}, B_{C_j}, C_{C_j}, 1_{C_j}, 2_{C_j}, 3_{C_j}$ and $A'_{C_j}, B'_{C_j}, C'_{C_j}, 1'_{C_j}, 2'_{C_j}, 3'_{C_j}$. Furthermore, we have links $\{\{x_i, 1_{x_i}\}, \{1_{x_i}, 2_{x_i}\}, \{2_{x_i}, 3_{x_i}\}, \{3_{x_i}, 4_{x_i}\}, \{4_{x_i}, 5_{x_i}\}, \{5_{x_i}, \bar{x}_i\} \mid i = 1, \dots, k\} \cup \{\{A_{C_j}, 1_{C_j}\}, \{1_{C_j}, l1_{C_j}\}, \{l1_{C_j}, B_{C_j}\}, \{B_{C_j}, 2_{C_j}\}, \{2_{C_j}, l2_{C_j}\}, \{l2_{C_j}, C_{C_j}\}, \{C_{C_j}, 3_{C_j}\}, \{3_{C_j}, l3_{C_j}\}, \{l3_{C_j}, A_{C_j}\} \mid l1_{C_j}, l2_{C_j}, l3_{C_j} \text{ the literals in } C_j, j = 1, \dots, l\}$ in the first graph G , the same links in the copy G' and $\{\{v, v'\} \mid v \in V[G]\}$ to connect them. We allow every two vertices to match and have the following preference lists. We only give the preferences for G , as preferences in G' are similar.

v	preferences	
x_i	$3_{x_i} \succ A_{C_1} \succ B_{C_1} \succ C_{C_1} \succ A_{C_2} \succ B_{C_2} \succ C_{C_2} \succ \dots \succ A_{C_l} \succ B_{C_l} \succ C_{C_l} \succ x'_i \succ \dots$	$i = 1, \dots, k$
\bar{x}_i	$3_{x_i} \succ A_{C_1} \succ B_{C_1} \succ C_{C_1} \succ A_{C_2} \succ B_{C_2} \succ C_{C_2} \succ \dots \succ A_{C_l} \succ B_{C_l} \succ C_{C_l} \succ \bar{x}'_i \succ \dots$	$i = 1, \dots, k$
1_{x_i}	$3_{x_i} \succ 1'_{x_i} \succ \dots$	$i = 1, \dots, k$
2_{x_i}	$3_{x_i} \succ 2'_{x_i} \succ \dots$	$i = 1, \dots, k$
3_{x_i}	$x_i \succ \bar{x}_i \succ 1_{x_i} \succ 5_{x_i} \succ 2_{x_i} \succ 4_{x_i} \succ 3'_{x_i} \succ \dots$	$i = 1, \dots, k$
4_{x_i}	$3_{x_i} \succ 4'_{x_i} \succ \dots$	$i = 1, \dots, k$
5_{x_i}	$3_{x_i} \succ 5'_{x_i} \succ \dots$	$i = 1, \dots, k$
A_{C_j}	$C_{C_j} \succ B_{C_j} \succ l1_{C_j} \succ 1_{C_j} \succ A'_{C_j} \succ \dots$	$l1_{C_j}$ 1. literal in $C_j, j = 1, \dots, l$
B_{C_j}	$A_{C_j} \succ C_{C_j} \succ l2_{C_j} \succ 2_{C_j} \succ B'_{C_j} \succ \dots$	$l2_{C_j}$ 2. literal in $C_j, j = 1, \dots, l$
C_{C_j}	$B_{C_j} \succ A_{C_j} \succ l3_{C_j} \succ 3_{C_j} \succ C'_{C_j} \succ \dots$	$l3_{C_j}$ 3. literal in $C_j, j = 1, \dots, l$
1_{C_j}	$A_{C_j} \succ 1'_{C_j} \succ \dots$	$j = 1, \dots, l$
2_{C_j}	$B_{C_j} \succ 2'_{C_j} \succ \dots$	$j = 1, \dots, l$
3_{C_j}	$C_{C_j} \succ 3'_{C_j} \succ \dots$	$j = 1, \dots, l$

Suppose the formula is satisfiable. We will give the matching edges within G . The same vertices are matched in G' (and edges are stable for the same reasons due to the similar preference lists). All vertices that remain unmatched that way are matched to their copy. At first, for every variable x_i we introduce $\{x_i, 3_{x_i}\}$ if x_i is true in the satisfying assignment, and $\{\bar{x}_i, 3_{x_i}\}$ otherwise. As the assignment is satisfying, every clause gadget has at least one of its literals matched. In case all three literals are matched, we generate the edges $\{A_{C_j}, 1_{C_j}\}, \{B_{C_j}, 2_{C_j}\}$ and $\{C_{C_j}, 3_{C_j}\}$. If $l1_{C_j}$ and $l2_{C_j}$ are matched or just $l2_{C_j}$ is matched, we match A_{C_j} to C_{C_j} and B_{C_j} to 2_{C_j} . The other cases of only one or two literals being matched are symmetric to this one. Now the remaining vertices are matched to their copies. We claim that this represents a locally stable matching.

Note that by the assignment edge, x_i respectively \bar{x}_i is matched to its favorite partner. In the case of 3_{x_i} being matched to \bar{x}_i , there is no chance for 3_{x_i} to learn about x_i . Hence all those edges are stable. Now, for clauses having all three literals satisfied, $1_{C_j}, 2_{C_j}$ and 3_{C_j} are matched to their favorite partner. For A_{C_j}, B_{C_j} and C_{C_j} the only vertex that yields an improvement and is accessible is already matched and not willing to switch. Thus, these edges are also stable. For the case of $l1_{C_j}$ and $l2_{C_j}$ or just $l2_{C_j}$ being matched, A_{C_j} and 2_{C_j} are matched to their first choice and B_{C_j} and C_{C_j} prefer their partner to all other unmatched vertices they know about and could convince to switch. This leaves us with the vertices matched to their copy. We note that every vertex prefers its copy to every other vertex in the other graph. Thus, a blocking pair can only arise with a vertex of its own graph and this vertex has to be matched to its copy as well. Now every 3_{x_i} as well as every A_{C_j}, B_{C_j} and every C_{C_j} is matched within its graph. But those are the only vertices possibly preferred to their copy by one of the vertices matched to their copy, that is, the edges between the copies are stable as well.

Now assume that the formula is unsatisfiable and M is a stable matching. First, we observe that none of the literals can be matched to some vertex A_{C_j} , as matching with $l1_{C_j}$ could be improved by matching with B_{C_j} (and B_{C_j} is always willing to match with A_{C_j}) and matching with every other literal can be improved by matching with 1_{C_j} , which again is always willing to switch to A_{C_j} . Similarly A_{C_j} is not matched outside the clause gadget, as it would switch to 1_{C_j} again. Similarly, B_{C_j} and C_{C_j} are not matched to any vertex outside their clause or any literal. As we cannot find a satisfying assignment, there has to be a clause gadget, such that none of its literals is matched to its vertex 3_{x_i} . Then A_{C_j} is not matched with 1_{C_j} because it could improve by switching to $l1_{C_j}$, because we know that no literal is matched with some $A_{C_{j'}}, B_{C_{j'}}$ or $C_{C_{j'}}$. For the same reasons $\{B_{C_j}, 2_{C_j}\} \notin M$ and $\{C_{C_j}, 3_{C_j}\} \notin M$. Then A_{C_j}, B_{C_j} and C_{C_j} have to be matched to higher valued vertices. But this implies that A_{C_j} is either matched to B_{C_j} or C_{C_j} , while B_{C_j} is either matched to A_{C_j} or C_{C_j} and C_{C_j} is either matched to A_{C_j} or B_{C_j} , that is, one vertex is involved in two matching edges. \square

Linear Polytope In addition to the computational complexity, we are also interested in structural characterizations of locally stable matchings. For the full information case, linear polytope representations have been shown to possess desirable properties, e.g., all extreme points being integer solutions in stable marriage or half-integrality for the roommates case (see [17]). Furthermore, every integer solution represents a feasible matching. Interestingly, we show that the latter property translates to locally stable matchings. This implies that whenever the LP has an integral solution, existence of a locally stable matching is guaranteed. The characterizations of extreme points form the basis of polynomial-time algorithms to decide existence of a stable matching [19]. We show below that the characterization of extreme points does not translate, which is consistent with our hardness result above.

Consider the following linear polytope with variables $x_e, e \in E$, and $x_v, v \in V$, and denote

$$N_e = (V, L \cup \{e\}).$$

$$\sum_{v \in e} x_e + x_v = 1 \quad \forall v \in V \quad (1)$$

$$\sum_{\substack{e'=\{u',v'\} \in E \\ u' \prec_{v'} u}} x_{e'} + x_{v'} + x_e \leq 1 \quad \forall v' \in V, e = \{u, v\} \in E, v \prec_u v', \text{dist}(u, v', N_e) \leq 2 \quad (2)$$

$$\sum_{\substack{e'=\{u',v'\} \in E, \\ u' \prec_{v'} u}} x_{e'} + x_{v'} + x_u \leq 1 \quad \forall v', u \in V, \text{dist}(u, v', N) \leq 2 \quad (3)$$

$$x_e, x_v \geq 0 \quad \forall e \in E, v \in V \quad (4)$$

For a matching M we define the *incident vector* x as $x_e = 1$, if $e \in M$, $x_v = 1$, if $v \in V \setminus \bigcup_{e \in M} e$ and $x_a = 0$ for all other $a \in V \cup E$.

Proposition 4. *The incidence vectors of locally stable matchings are precisely the integer solutions of the polytope (1)-(4).*

Proof. First, (1) and (4) can be satisfied by an integer vector x iff x is the incident vector of a matching. Now let M be a locally stable matching and x its incident vector. Assume that x violates one of the equations (2) or (3). This is only possible (for an integer vector) for (2) if $x_{e=\{u,v\}} = 1$ and at least one $x_s = 1$ for some $s = \{u', v'\} \in E, u' \prec_{v'} u$, or $s = v'$. But then $\{u, v'\}$ is a local blocking pair in M in contradiction to M being locally stable. Similarly for (3) x_u and some $x_s, s = \{u', v'\} \in E, u' \prec_{v'} u$, or $s = v'$, have to be one, which again lead to the local blocking pair $\{u, v'\}$ in M .

Conversely let x be an integer solution of (LP2) and M the matching defined by x . Assume that $\{u, v'\}$ is a local blocking pair in M . If u is matched through e and v' is visible for u in N_e then

$$\sum_{\substack{e'=\{u',v'\} \in E, \\ u' \prec_{v'} u}} x_{e'} + x_{v'} + x_e = 2.$$

The same holds for the roles of u and v' switched. Note that these are the only two possibilities for u and v' to see each other through the use of a matching edge. If they both are unmatched,

$$\sum_{\substack{e'=\{u',v'\} \in E, \\ u' \prec_{v'} u}} x_{e'} + x_{v'} + x_u = x_{v'} + x_u = 2$$

Hence, there are no local blocking pairs for M . □

Unlike the inequalities given in [17] for stable matchings, the polytope above does not define a solution set where all extreme points are integer solutions in the bipartite case.

Example 1. Consider the network with $U = \{1, 2\}$, $W = \{A, B, C, D\}$, $L = \{\{1, 2\}, \{A, B\}, \{A, C\}, \{B, D\}, \{C, D\}\}$ and $E = \{\{i, j\} | i \in U, j \in W\}$. Now A and B prefer 2 to 1 and C and D 1 to 2. 1 ranks D over B over C over A and 2 ranks B over D over A over C . This way the only two locally stable matchings are $\{\{1, B\}, \{2, D\}\}$ and $\{\{1, D\}, \{2, B\}\}$, but the fractional matching $\{\epsilon\{1, A\}, (\frac{1}{2} - \epsilon)\{1, B\}, \frac{1}{2}\{1, D\}, \frac{1}{2}\{2, B\}, \frac{1}{2}\{2, D\}\}$ is in the solution set of the associated polytope.

One might think that this problem could be prevented if we set edges directly to 0 in case they never appear in a locally stable matching. However, if we extend our example by a third vertex

$3 \in U$, allow the matching edge $\{3, B\}$ and make 3 B 's favorite partner, there are locally stable matchings that use $\{1, A\}$, but our given solution is still not a linear combination of the integer solutions.

References

- [1] Heiner Ackermann, Paul Goldberg, Vahab Mirrokni, Heiko Röglin, and Berthold Vöcking. Uncoordinated two-sided matching markets. *SIAM Journal on Computing*, 40(1):92–106, 2011.
- [2] Kemal Akkaya, Ismail Guneydas, and Ali Bicak. Autonomous actor positioning in wireless sensor and actor networks using stable-matching. *Intl. J. Parallel, Emergent and Distrib. Syst.*, 25(6):439–464, 2010.
- [3] Esteban Arcaute and Sergei Vassilvitskii. Social networks and stable matchings in the job market. In *Proceedings of the 5th International Workshop on Internet and Network Economics*, WINE '09, pages 220–231, Berlin, Heidelberg, 2009. Springer-Verlag.
- [4] P. Austrin, S. Khot, and M. Safra. Inapproximability of vertex cover and independent set in bounded degree graphs. In *Computational Complexity, 2009. CCC '09. 24th Annual IEEE Conference on*, pages 74 –80, july 2009.
- [5] M.X. Goemans, Li Li, V.S. Mirrokni, and M. Thottan. Market sharing games applied to content distribution in ad hoc networks. *Selected Areas in Communications, IEEE Journal on*, 24(5):1020 – 1033, may 2006.
- [6] Dan Gusfield and Robert Irving. *The Stable Marriage Problem: Structure and Algorithms*. 1989.
- [7] Magnus Halldórsson, Robert Irving, Kazuo Iwama, David Manlove, Shuichi Miyazaki, Yasufumi Morita, and Sandy Scott. Approximability results for stable marriage problems with ties. *Theoretical Computer Science*, 306(1–3):431–447, 2003.
- [8] Martin Hoefer. Local matching dynamics in social networks. In Luca Aceto, Monika Henzinger, and JirÅ Sgall, editors, *Automata, Languages and Programming*, volume 6756 of *Lecture Notes in Computer Science*, pages 113–124. Springer Berlin / Heidelberg, 2011.
- [9] Robert W Irving. An efficient algorithm for the "stable roommates" problem. *Journal of Algorithms*, 6(4):577 – 595, 1985.
- [10] Alexander Kelso and Vincent Crawford. Job matchings, coalition formation and gross substitute. *Econometrica*, 50(1483–1504), 1982.
- [11] Zoltán Király. Better and simpler approximation algorithms for the stable marriage problem. *Algorithmica*, 60(1):3–20, 2011.
- [12] Donald Knuth. *Marriages stables et leurs relations avec d'autres problemes combinatoires*. Les Presses de l'Université de Montréal, 1976.
- [13] Fabien Mathieu. Self-stabilization in preference-based systems. *Peer-to-Peer Netw. Appl.*, 1(2):104–121, 2008.

- [14] Eric McDermid. A $3/2$ -approximation algorithm for general stable marriage. In Susanne Albers, Alberto Marchetti-Spaccamela, Yossi Matias, Sotiris Nikolettseas, and Wolfgang Thomas, editors, *Automata, Languages and Programming*, volume 5555 of *Lecture Notes in Computer Science*, pages 689–700. Springer Berlin / Heidelberg, 2009.
- [15] Alvin Roth and Marilda Oliveira Sotomayor. *Two-sided Matching: A study in game-theoretic modeling and analysis*. 1990.
- [16] Alvin Roth and John Vande Vate. Random paths to stability in two-sided matching. *Econometrica*, 58(6):1475–1480, 1990.
- [17] Alvin E. Roth, Uriel G. Rothblum, and John H. Vande Vate. Stable matchings, optimal assignments, and linear programming. *Mathematics of Operations Research*, 18(4):pp. 803–828, 1993.
- [18] Alvin E. Roth, Tayfun Sönmez, and M. Utku Ünver. Pairwise kidney exchange. *Journal of Economic Theory*, 125(2):151 – 188, 2005.
- [19] Chung-Piaw Teo and Jay Sethuraman. The geometry of fractional stable matchings and its applications. *Mathematics of Operations Research*, 23(4):pp. 874–891, 1998.
- [20] H. Yanagisawa. *Approximation algorithms for stable marriage problems*. PhD thesis, Kyoto University, Graduate School of Informatics, 2007.